(67686) Mathematical Foundations of AI	July 30, 2008
Lecture 11	
Lecturer: Ariel D. Procaccia	*Scribe: Michael Zuckerman and Na'ama Zohary*

1 Cooperative Games

$N = \{1, \ldots, n\}$ is the set of players (agents). $v : 2^N \to \mathbb{R}$ is the function, which gives a value for every coalition (subset) of agents.

There are two schools in cooperative games research:

1. With Transferable Utility - studying settings where money can be freely passed inside the coalitions, and

2. Without Transferable Utility - there are predefined payment vectors which can be payed to the agents, and no other payments are allowed.

We will concentrate on the first setting.

Definition 1 *Function v is super-additive if for every $B, C \subseteq N$ such that $B \cap C = \emptyset$, $v(B \cup C) \geq v(B) + v(C)$.*

If we assume that v is super-additive, then we assume that the grand coalition always forms, and the question is how to divide the value $v(N)$ between the agents.
Example $N = \{1, 2, 3\}$
$v(\emptyset) = v(1) = v(2) = v(3) = 0$
$v(\{1, 2\}) = v(\{1, 3\}) = v(\{2, 3\}) = v(N) = 1$

Denote by $\overrightarrow{x} = \langle x_1, \ldots, x_n \rangle$ the vector of payments s.t. $\sum_{i \in N} x_i = v(N)$. For $B \subseteq N$ we will denote $x(B) = \sum_{i \in B} x_i$.

Definition 2 *We say that payoff vector $\overrightarrow{x}$ is blocked by coalition B if $v(B) > x(B)$. The* core *of the game contains all the vectors which are not blocked by any coalition.*

Example There are games with empty core, such as the game in Example 1: if by way of contradiction the core of this game is not empty, let $\overrightarrow{x}$ be a vector in the core. $x(N) = 1$ and so there is $i \in N$ s.t. $x_i > 0$. W.l.o.g. $x_1 = \epsilon > 0$. Then $x(\{2, 3\}) = 1 - \epsilon < 1 = v(\{2, 3\})$, and so the coalition $\{2, 3\}$ is blocking, a contradiction.

2 Determining Nonemptiness of the Core

The question whether the core of a given game is empty is an important computational question.

2.1 Compact representation of cooperative game

N is represented as before, and also we have the set of pairs $W = \{(B, v(B))\}$ where each pair contains a coalition $B \subseteq N$ and its value $v(B)$.

$$v(B) = \max\{\sum_{j=1}^{r} v(B_j) : B_1, \dots, B_r \text{ is a partition of } B, \forall j (B_j, v(B_j)) \in W\}$$

The problem of computing $v(B)$ can be computationally hard.
Example $N = \{1, 2, 3\}$
$W = \{(\{1\}, 3), (\{2\}, 1), (\{3\}, 0)\}$
$v(\{1, 2\}) = v(\{1\}) + v(\{2\}) = 4$

Definition 3 *In the* CORE-NONEMPTY *problem we are given a cooperative game in the above representation, and the question is whether the core is non-empty.*

Definition 4 *In the* X3C *problem we are given a set U and a family $S_1, S_2, \dots$ of subsets of U, where each $S_j \subseteq U$, $|S_j| = 3$, and also $|U| = 3m$. The question is: are there m subsets S_j that cover the set U.*

Fact 5 *X3C is $\mathcal{NP}$-hard.*

Theorem 6 *The CORE-NONEMPTY problem is $\mathcal{NP}$-hard.*

Proof We prove the theorem by providing a reduction from X3C problem. Given as input X3C we will build the cooperative game: $N = (U \cup \{a, b\})$. The set W will contain the pair $(S_j, 3)$ for each subset S_j from the input X3C. In addition W contains $(U \cup \{a\}, 6m)$, $(U \cup \{b\}, 6m)$ and $(\{a, b\}, 6m)$. Finally, $\forall i \in U$ $(\{i\}, 0) \in W$ and also $(\{a\}, 0)$, $(\{b\}, 0) \in W$. We will show that there is an exact cover in X3C if and only if the core of the cooperative game is non-empty.

$\Rightarrow$ There exists a partition of U into m pairwise disjoint subsets S_j, and since for all j $(S_j, 3) \in W$, it holds $v(U) = 3m$. We can partition the set N into U and $\{a, b\}$, and so $v(N) \geq 9m$. Now we build the payment vector. $\forall i \in U$ $x_i = 1$, $x_a = 3m$, $x_b = 3m$. It is easy to see that no coalition can gain by deviation from the grand coalition.

$\Leftarrow$ Let's assume that there is no exact cover in X3C problem. Now $v(U) < 3m$, and hence $v(N) < 9m$. Now examine U, a and b. If all of them get less than $3m$, then every pair is blocking, because it can leave the grand coalition and get $6m$. If w.l.o.g. $x_a > 3m$, then $x_{U \cup \{b\}} < 6m$, and so $U \cup \{b\}$ is blocking. ∎

2.2 Information About the Grand-Coalition Makes the Problem Tractable

In the last section we saw that when the data on the game is compact, it made the problem of determining the nonempiness of the core hard. (If the data was not compact but exponential, then it would not be a problem to have an algorithm that is polynomial in that data.)
Remember that in the concise representation:

$$\forall B \quad v(B) = \max_{\{B_j\}} \{\sum_{j=1}^{r} v(B_j) | \ (\{B_j\}_{j=1}^{r} \textit{ is a partition of } B) \ \wedge \ (\forall 1 \leq j \leq r \ (B_j, v(B_j)) \in W)\}$$

This means that calculating the value of the grand coalition (N) may be hard in its self. The next claim will show that the hardness of CORE-NONEMPTY was due to the difficulty in determining what the grand coalition can accomplish. i.e. when $v(N)$ is explicitly provided the problem becomes easy.

Claim 7 *When $v(N)$ is explicitly provided (in addition to W), CORE-NONEMPTY is in P.*

Proof

Lemma 8 *for every payment vector $\overrightarrow{x}$:*

$$(\exists \text{ coalition } C \subseteq N \text{ blocking } \overrightarrow{x}) \Leftrightarrow (\exists \text{ coalition } B \text{ blocking } \overrightarrow{x} \text{ such that } (B, v(B)) \in W)$$

Proof of Lemma :

$\Leftarrow$ If there exists coalition B blocking $\overrightarrow{x}$ such that $(B, v(B)) \in W$
then obviously B is a coalition in N blocking $\overrightarrow{x}$.

$\Rightarrow$ Assuming there exists a coalition $C \in N$ blocking $\overrightarrow{x}$. i.e. $v(C) > x(C)$. Then:
$\exists \{B_j\}_{j=1}^{r}$ partition of C such that $\forall 1 \leq j \leq r \ (B_j, v(B_j)) \in W \ \wedge \ v(C) = \sum_{j=1}^{r} v(B_j)$.

And $x(C) = \sum_{j=1}^{r} x(B_j)$ (because the payment is individual and the individuals composing $\{B_j\}_{j=1}^{r}$ are exactly the individuals in C)

Therefore: $\sum_{j=1}^{r} v(B_j) = v(C) > x(C) = \sum_{j=1}^{r} x(B_j)$

Since the first sum is definitely greater than the second, at least one item in the first is strictly greater than one item in the second. i.e. there exists a $j \in \{1, \ldots, r\}$ such that $v(B_j) > x(B_j)$.

Therefore, $\exists B_j$ such that $(B_j, v(B_j)) \in W$ and B_j is blocking.

■

Following the lemma, for each $\overrightarrow{x}$, $\overrightarrow{x}$ is not blocked if it is not blocked by any B specified in W. The core is not empty if we can find such a payment vector. Therefore, the problem that needs to be solved is:

Find $\overrightarrow{x} = < x_1, \ldots, x_n >$ such that

1. $\sum_{i \in N} x_i = v(N)$ - the payment vector is valid
2. $\forall (B, v(B)) \in W \quad v(B) \leq x(B) (= \sum_{i \in B} x_i)$ - there are no blocking coalitions in W

This is an example of a Linear Program. Linear Programs can be solved in Polynomial time in the number of variables and constraints. In our case, the variables are $x_1, \ldots, x_n$ and there are $|W|+1$ constraints. Note that we need $V(n)$ to define the first constraint. The Linear Program will not only specify if the core is not empty, but also provide a $\overrightarrow{x} \in CORE$ if one exists. ■

(A brief summary on Linear Programming appears in Appendix A)

References

[1] Conitzer V. and Sandholm T. 2006. Complexity of Constructing Solutions in the Core Based on Synergies Among Coalitions. *Artificial Intelligence, 170: 607-619*

3 Determining Nonemptiness of the Core in Simple Games

3.1 Definitions

Definition 9 (Simple Games) *A Coalitional Game is Simple if*

1. *for every $B \subseteq N$ either $v(B) = 0$ or $v(B) = 1$ $(V : 2^n \rightarrow \{0, 1\})$ and*
2. $\forall B, C \subseteq N \quad (v(B) = 1 \ \wedge \ B \subseteq C) \ \rightarrow \ (v(C) = 1)$

Remark Such games are demonstrated when players are members of a board. In such games, a proposed bill or decision is either passed or rejected. Those subsets of the players that can pass bills without outside help are called winning coalitions while those that cannot are called losing coalitions. Typical examples of simple games are

1. the majority rule game where v(S) = 1 if $|S| > \frac{1}{2}n$, and v(S) = 0 otherwise;
2. the unanimity game where v(S) = 1 if S = N and v(S) = 0 otherwise;
3. the dictator game where v(S) = 1 if $i^* \in S$ and v(S) = 0 otherwise.

Assumption 10 *We will always assume that v(N) is 1. (Otherwise, there is no use in cooperation because no coalition wins.)*

Remark Notice that Simple Games are not always Super-Additive.

Example
When there are two disjoint coalitions that win separately, then their union's worth is less then their sum.
$N = a, b \quad v(a) = 1 \ v(b) = 1 \quad v(a, b) = 1 < 2 = (v(a) + v(b))$

Definition 11 (Proper Simple Games)
A Simple Game is Proper if $\forall B \subseteq N \quad (v(B) = 1) \ \rightarrow \ (v(N \setminus B) = 0)$

Claim 12 *If a Simple Game is Proper, then it is Superadditive*

Proof
for every $B, C \subseteq N$:

- If the value of one (at least) of the coalitions is 0, without loss of generality $V(B) = 0$ then

  1. $v(B) + v(C) = v(C)$.
  2. $v(C) \leq v(B \cup C)$ since $(v(c) = 1 \rightarrow \; v(B \cup C) = 1)$

  $$\text{Therefore: } v(B) + v(C) \leq v(B \cup C)$$

- Else, $v(B) = 1 \;\wedge\; v(C) = 1$. Since the game is proper, $B \cap C \neq \emptyset$ (because if $B \cap C = \emptyset$ then one set would be in the complement of the other and its value would be 0). Therefore, superadditivity applies because it is relevant only to disjoint subsets.

■

3.2 Determining Nonemptiness of the Core in Simple Games

A simple game can be represented concisely by the set of players N and the set of all minimal winning coalitions: $W = \{B | \; (B \subseteq N) \;\wedge\; (v(B) = 1) \;\wedge\; (\neg \exists C \; C \subset B \;\wedge\; v(C) = 1)\}$.
This way, for every $C \subseteq N$ $[v(C) = 1] \Leftrightarrow [\exists B \; (B \subseteq C) \;\wedge\; B \in W)]$.
Note that, as before, this representation could still be exponential: $\binom{n}{\frac{1}{2}n}$ in the worst case. However, in most cases, it would be considerably less.

Definition 13 (Veto Player) *In a simple game, (N, v), a player, i^*, is said to be a Veto player, if i^* is in every winning coalition: $\forall B \subseteq N \; (v(B) = 1) \;\rightarrow\; (i^* \in B)$.*

Claim 14 *An equivalent definition is: a player i^* is a veto player if $v(N \setminus i^*) = 0$.*

Proof

$\Leftarrow$ if i is a veto player then obviously $v(N \setminus i) = 0$.

$\Rightarrow$ if i isn't a veto player then there exists a coalition $B \subseteq N$ $i \notin B$ such that $v(B) = 1$. Since $B \subseteq N \setminus \{i\}$, $v(N \setminus i) = 1$

■

Claim 15 *The core of a Simple Game is not empty if and only if there is (at least one) veto player in N.*

Remark The game does not have to be proper for this to be true

Proof

⇒ If there is a veto player i^* we define the payment $\overrightarrow{x}$ so that all the reward goes to the veto player:

$$x_i = \begin{cases} v(N)(=1) & (i = i^*) \\ 0 & (i \neq i^*) \end{cases}$$

We'll show that no coalition blocks $\overrightarrow{x}$. For every coalition B:

- if $i^* \notin B$ then $v(B) = 0$ (because it doesn't include a veto player) therefore $v(B) = x(B)$ and B isn't blocking.
- if $i^* \in B$ then $x(B) = 1$ (because it includes i^*) therefore $v(B) \leq 1 = x(B)$ and B isn't blocking.

⇐ If the core isn't empty, let us look at a $\overrightarrow{x} \in CORE$:
Since $(x(N) = v(N) = 1)$ and $(x(N) = \sum_{i \in N} x_i)$, $\exists$ player i^* such that $x_{i^*} = \epsilon > 0$.
Observe coalition: $N \setminus \{i^*\}$: since it isn't blocking: $v(N \setminus \{i^*\}) \leq x(N \setminus \{i^*\}) = 1 - \epsilon < 1$.
Therefore, because this is a Simple game, $v(N \setminus \{i^*\}) = 0 \rightarrow$ coalition $N \setminus \{i^*\}$ losses.
From the equivalent definition, i^* is a veto player $\rightarrow$ there is a veto player in the game.

■

Corollary 16 *Determining CORE-NONEMPTY in Simple Games (given input W) is in P.*

Proof Following the claim above, in order to determine if the core is empty, we need to see if there aren't any veto players. Remember that W holds all the minimal winning coalitions. Therefore, for every player i, i isn't a veto player if and only if there exists a coalition in W that doesn't include i (because if there is a winning coalition that doesn't include i then there is a minimal one in W).
So, the determining algorithm will go over all the groups in W and check, for every player, if it appears in the group. If there is a group that doesn't include a certain player, then that player isn't a veto player ($\forall i \quad (\exists B \in W \ i \notin B) \rightarrow (i \ isn't \ a \ veto \ player)$).
Therefore, with the right data-structure, the complexity of this algorithm is O($|W| * n$).
Finding the veto players will also give us a $\overrightarrow{x} \in CORE$ if the core isn't empty. ■

Remark [Characterization of the core]
The core of a Simple Game is all the payment vectors that divide $v(N)(= 1)$ amongst the veto players and give no payment to any other player.
i.e. $CORE = \{\overrightarrow{x} | \ [\forall i \ (x_i > 0) \ \rightarrow \ (i \ is \ a \ veto \ player)] \ \wedge \ [(\sum_{i \ a \ veto \ player} x_i) = 1]\}$.
Notice that the above proof is adequate whether there is one veto player or more.
From the second part (⇒) we see that for any $\overrightarrow{x}$ in the core, $x_i > 0$ only for i-s that are veto players. And the first part (⇐) works the same for any $\overrightarrow{x}$ as described here.

Remark In coalitional games the payments' purpose is to give incentive for cooperation. As we can see, they are not always envy-free. A non-veto player can definitely envy a veto player's payment. However, since the non-veto player has much less power, there is nothing he can do about it.

A Linear Programs

A.1 Definition of an LP

Linear Programming (LP) problems involve the optimization of a linear objective function ($f(x) = \sum_i a_i x_i)$), subject to linear equality and inequality constraints.
Informally, the vector x is in an n dimensional space. Each constraint defines a line through this space. The linear program returns the point that minimizes or maximizes the objective function in the space bounded by the lines (a polytope).
The problem can be expressed in canonical form:
Maximize $c^T x$ subject to
$Ax \leq b$
$x = (x_1, \ldots, x_n)$ is the vector of variables. $c \in F^n$ is a vector of coefficients of the objective function. $b \in F^m$ and $A \in F^{m \times n}$ hold the coefficients of the constraints.

A.2 Solving an LP

The Simplex method was published in 1947 by George B. Dantzig. This algorithm is based on the fact that, if there exists a feasible solution to the problem, it is on the edges of the polytope. The algorithm walks along the edges always moving to the the next best neighbor. Although this algorithm is quite efficient in practice and can be guaranteed to find the global optimum if certain precautions against cycling are taken, it has poor worst-case behavior: it is possible to construct a linear programming problem for which the simplex method takes a number of steps exponential in the problem size. In fact, for some time it was not known whether the linear programming problem was solvable in polynomial time.
This long standing issue was resolved by Leonid Khachiyan in 1979 with the introduction of the ellipsoid method, the first worst-case polynomial-time algorithm for linear programming. Basically the algorithm envelopes the polytope in an ellipsoid. At each iteration the volume of the ellipsoid shrinks by a constant factor, at each step making certain to still retain the optimal vertex.
The algorithm had little practical impact, as the simplex method is more efficient for all but specially constructed families of linear programs. An even larger major theoretical and practical breakthrough in the field came in 1984 when Narendra Karmarkar introduced a new interior point method for solving linear programming problems. Karmarkar's algorithm not only improved on Khachiyan's theoretical worst-case polynomial bound, but also promised dramatic practical performance improvements over the simplex method.

Taken from: http://en.wikipedia.org/wiki/Linear-programming